Boundary Detection

https://liaoux.github.io/uir-cv-2026/

Introduction to Computer Vision
Spring 2026, Lecture 4

Today: Boundary Detection

We need to find Object Boundaries from Edge Pixels.

Topics:

1. Fitting Lines and Curves to Edges.

Boundary Detection

We need to find Object Boundaries from Edge Pixels.

Topics:

1. Fitting Lines and Curves to Edges.
2. Active Contours (Snakes).

Preprocessing Edge Images

Real Vase

Object Boundaries

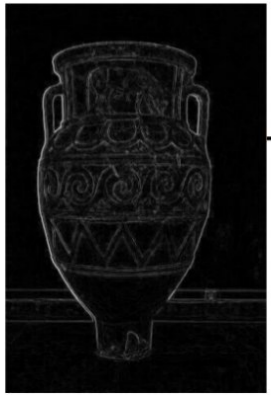
Preprocessing Edge Images

Real Vase

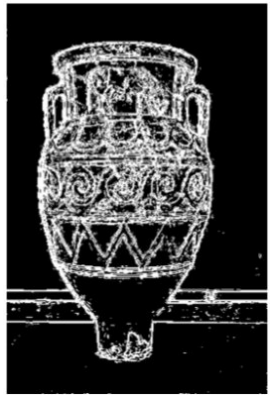
Edge Detection

Ex: Sobel

Gradient Magnitude



Thresholding



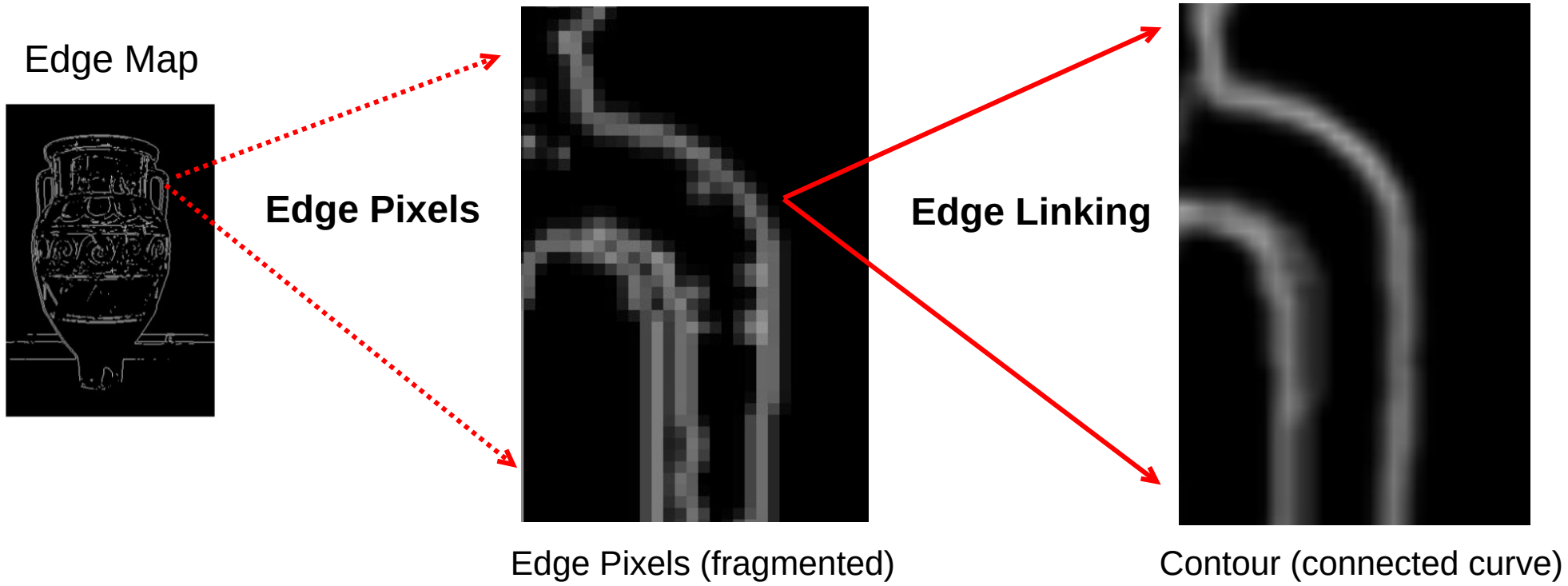
Thinning

Ex: NMS

Edge Linking

Object Boundaries

From Edge Pixels to Object Contours

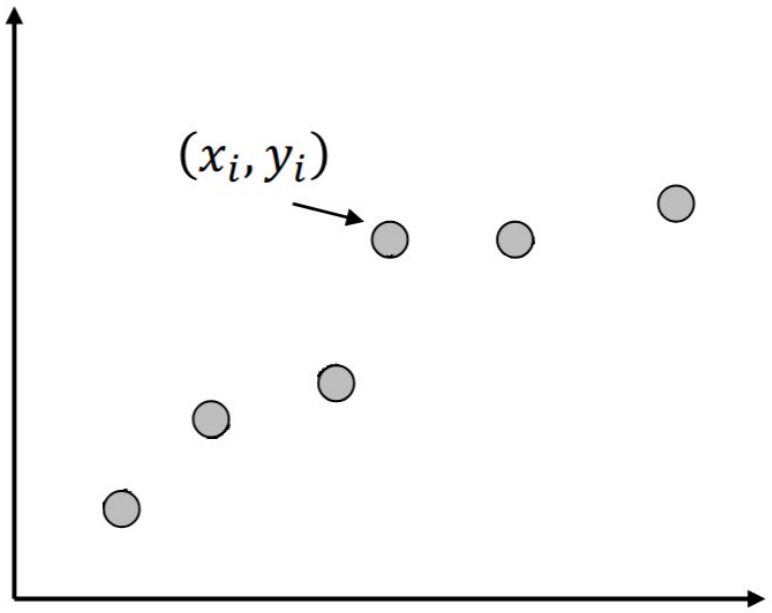


Goal: Boundary detection links edge pixels into continuous contours that represent the outlines of objects.

Fitting Lines and Curves

Represent Edge Pixels as Points

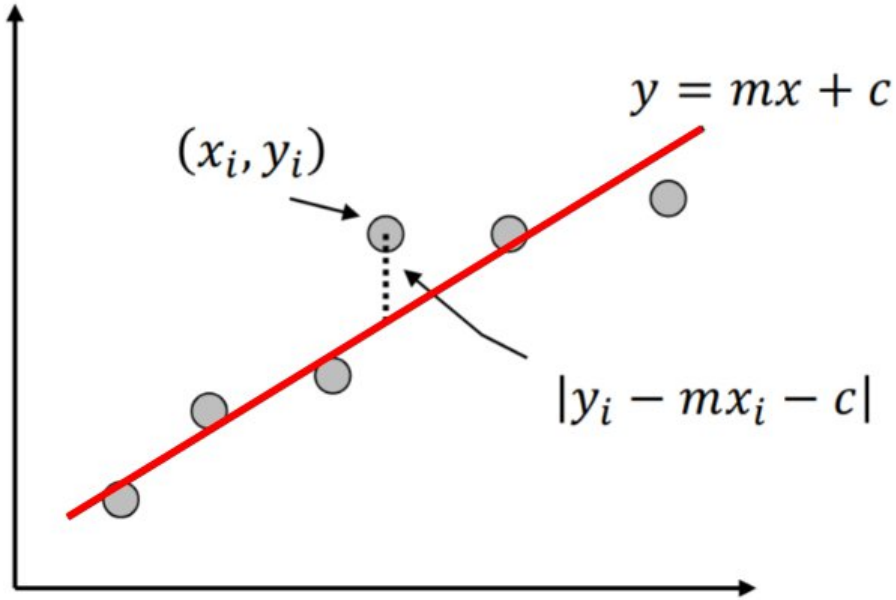
- After edge detection, we obtain a **set of edge pixels** represented as points (x_i, y_i)
- These pixels often lie along the **boundaries of objects**.
- To represent these boundaries, we approximate the edge points using **geometric models**:
  - **Straight lines** for linear edges
  - **Curves** for non-linear object boundaries
- This modeling helps convert **noisy pixel data into meaningful shapes**.



Fitting Lines to Edges

Given: Edge Points (x_i, y_i)

Task: Find the **slope** m and **intercept** c



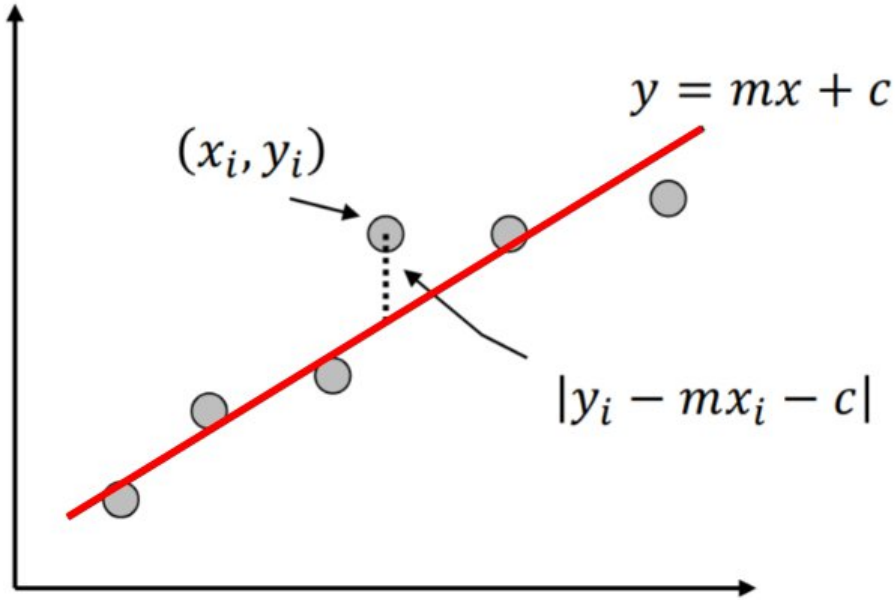
Fitting Lines to Edges

Given: Edge Points (x_i, y_i)

Task: Find the **slope** m and **intercept** c

Minimize: Average Squared Vertical Distance

$$E = \frac{1}{N}\sum_i (y_i - mx_i - c)^2$$

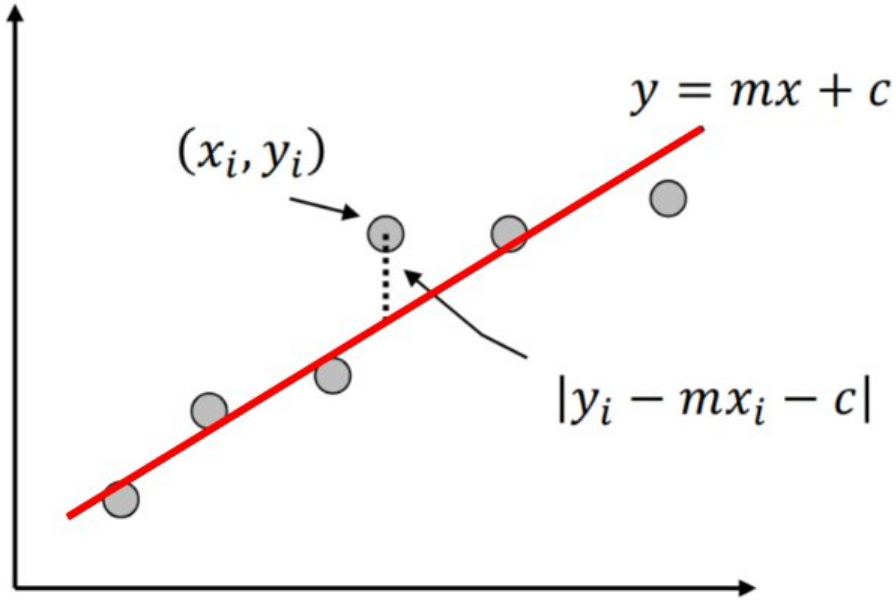


Fitting Lines to Edges

Given: Edge Points (x_i, y_i)

Task: Find the **slope** m and **intercept** c

Minimize: Average Squared Vertical Distance



$$E = \frac{1}{N}\sum_i (y_i - mx_i - c)^2$$

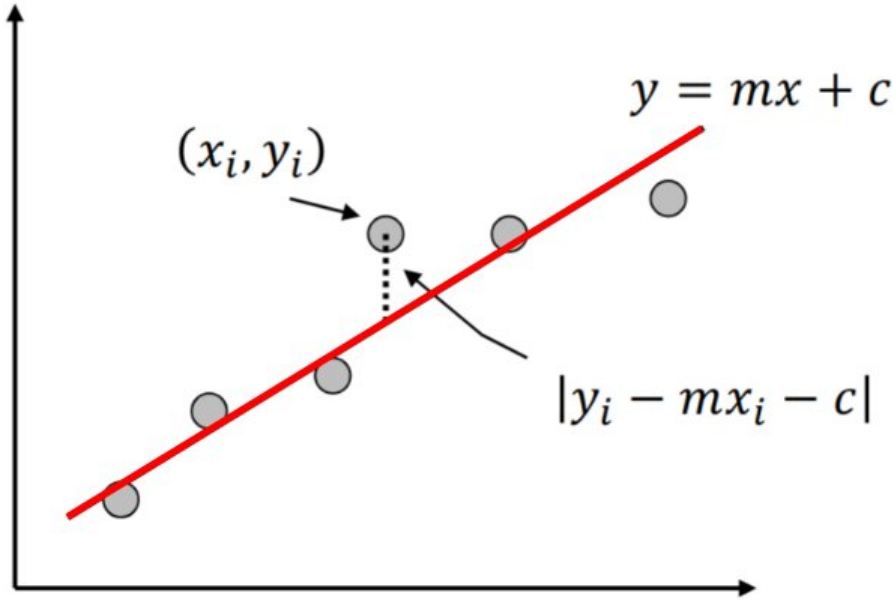
Least Squares Solution:

$$\frac{\partial E}{\partial m} = \frac{-2}{N}\sum_i x_i(y_i - mx_i - c) = 0 \qquad \frac{\partial E}{\partial c} = \frac{-2}{N}\sum_i (y_i - mx_i - c) = 0$$

Fitting Lines to Edges

Given: Edge Points (x_i, y_i)

Task: Find the **slope** m and **intercept** c



Solution:

$$m = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2} \qquad c = \bar{y} - m\bar{x}$$

where: $\bar{x} = \frac{1}{N}\sum_i x_i \qquad \bar{y} = \frac{1}{N}\sum_i y_i$

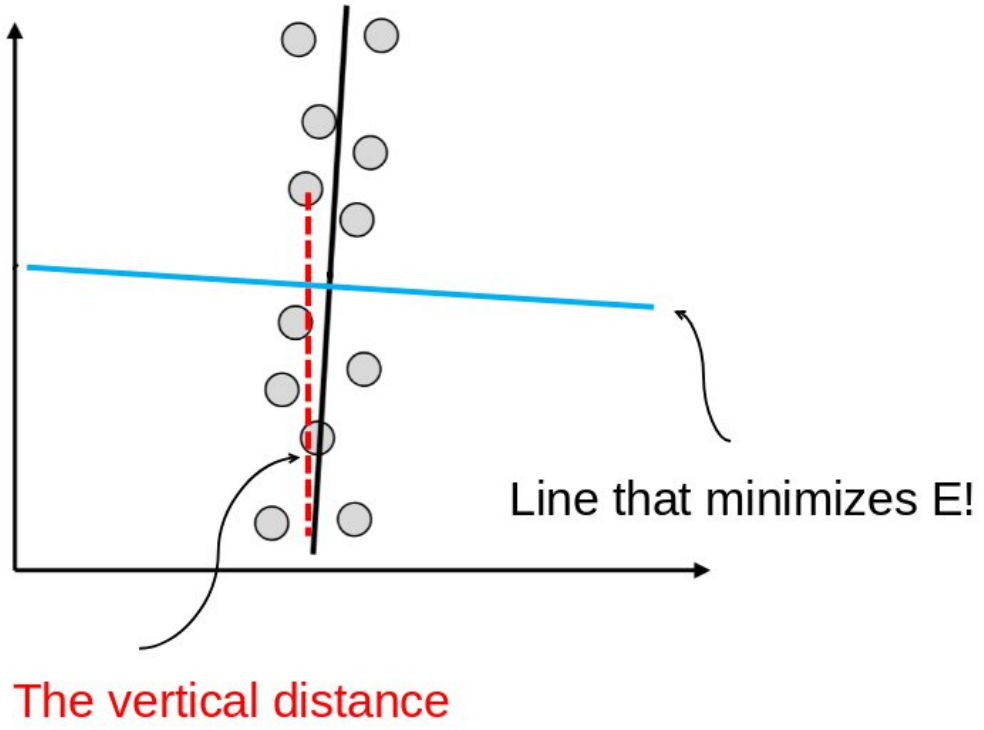
Closed form solutions

Fitting Lines to Edges

Problem: When the points (x_i,y_i) represent a vertical line.

Cause: We were minimizing a cost function that represents the vertical distance of each point from the line.

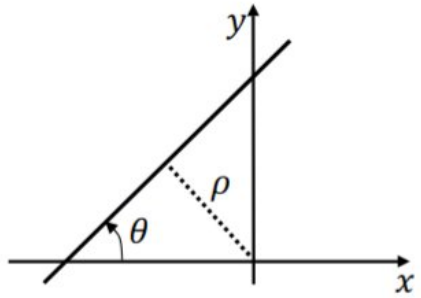
Note: The vertical distance from any of these points to the desired vertical line is fairly large.



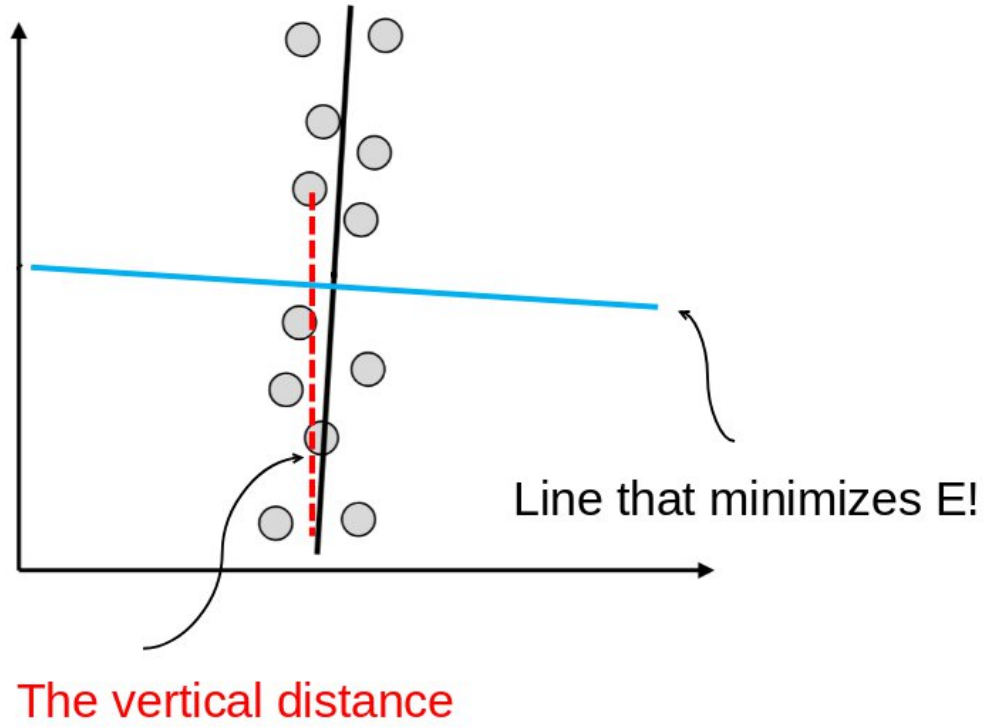
Fitting Lines to Edges

Problem: When the points (x_i, y_i) represent a vertical line.

Solution: Use a different line equation



$$x \sin\theta - y \cos\theta + \rho = 0$$



- ρ (rho) is the shortest distance from the line to the origin.
- θ *(theta)* is the angle that the line makes with respect to the horizontal axis.

Fitting Lines to Edges

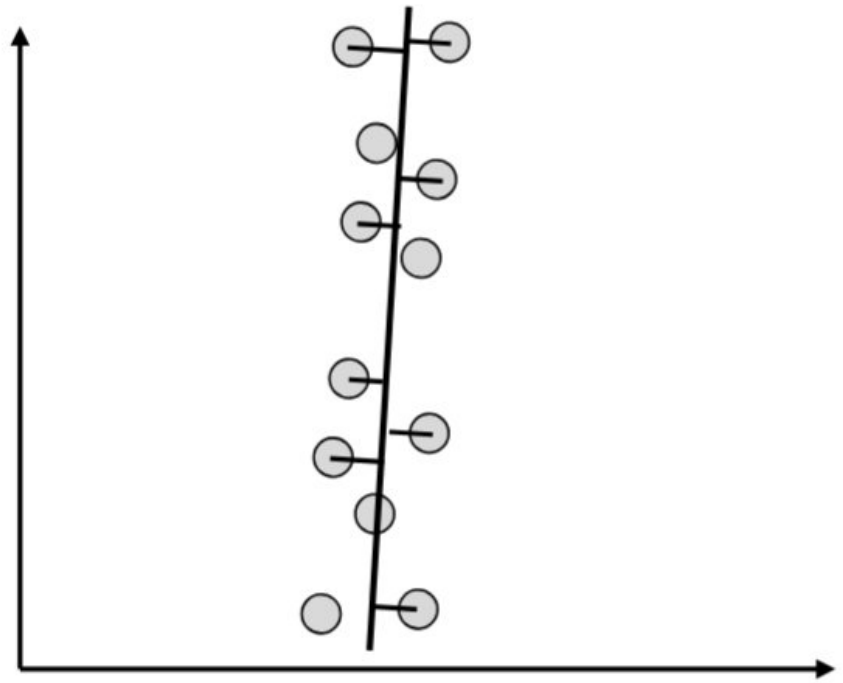
The general distance from point (x_i, y_i) to line $ax + by + c = 0$ is:

$$r_i = \frac{|ax_i + by_i + c|}{\sqrt{a^2 + b^2}}$$

Where in normal form:

- $a = \sin\theta, b = \cos\theta, c = \rho$
- Normalizer $\sqrt{sin^2\theta + cos^2\theta} = 1$

so $r_i = |ax_i + by_i + c|$ LHS of normal equation!



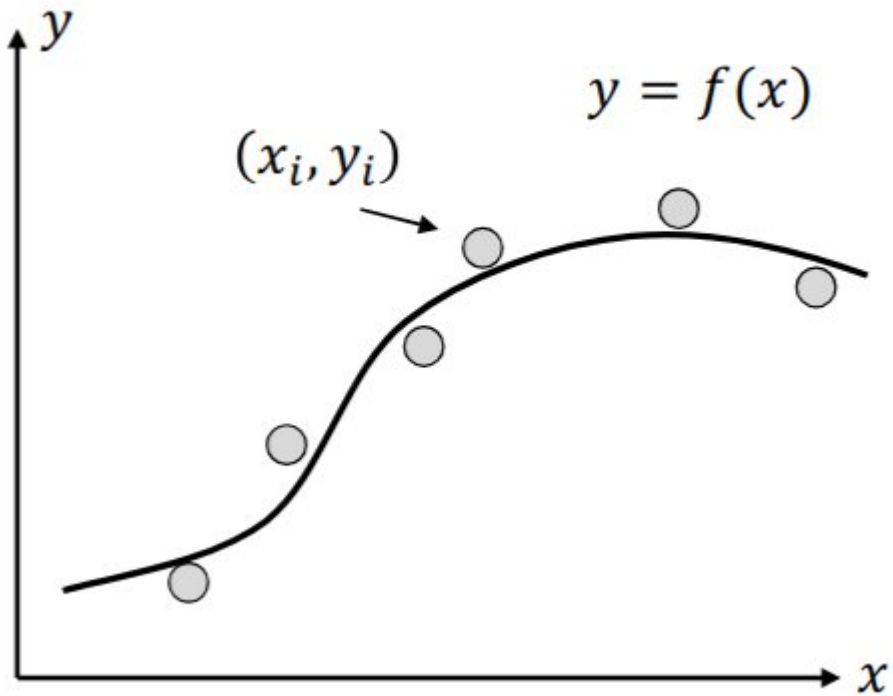
Fitting Curves to Edges

Given: Edge points (x_i, y_i)

Task: Find polynomial $y = ax^3 + bx^2 + cx + d$ that best fits the points.

Minimize: Average Squared Vertical Distance

$$E = \frac{1}{N}\sum_i (y_i - ax_i^3 - bx_i^2 - cx_i - d)^2$$



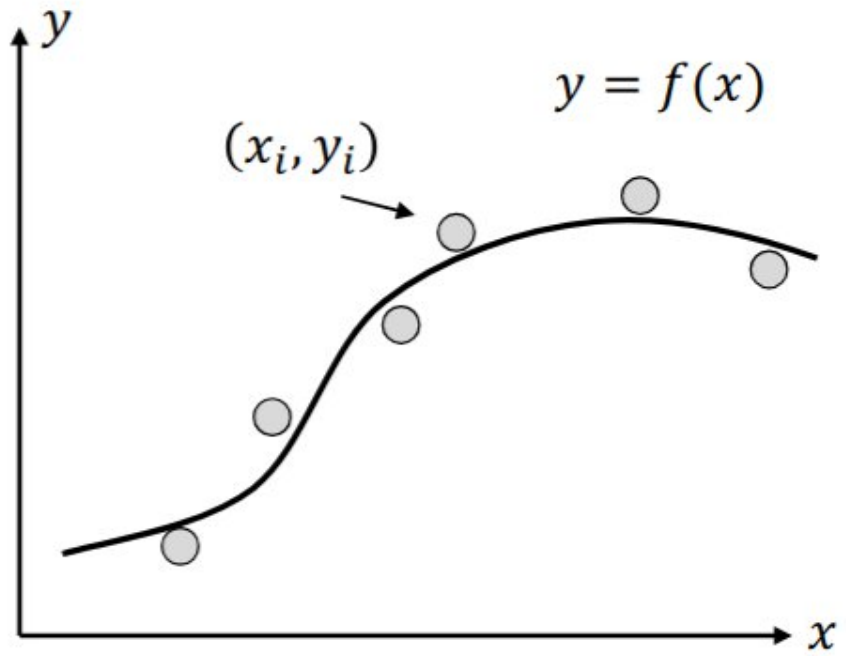
Solve the Linear System Using Least Squares Fit by:

$$\frac{\partial E}{\partial a} = 0 \quad \frac{\partial E}{\partial b} = 0 \quad \frac{\partial E}{\partial c} = 0 \quad \frac{\partial E}{\partial d} = 0$$

Fitting Curves to Edges

Solving as a Linear System:

$$y_0 = ax_0^3 + bx_0^2 + cx_0 + d$$
$$y_1 = ax_1^3 + bx_1^2 + cx_1 + d$$
$$\vdots$$
$$y_i = ax_i^3 + bx_i^2 + cx_i + d$$
$$\vdots$$
$$y_n = ax_n^3 + bx_n^2 + cx_n + d$$



Given many (x_i, y_i)'s, this is an over-determined linear system with four unknowns (a, b, c, d).

MATH PRIMER: Solving a Linear System

An over-determined linear system with m unknowns $\{a_j\}$ $(j = 0, \dots, m)$ and n observations $\{(x_{ij}, y_i)\}$ $(i = 0, \dots, n)$ $(n > m)$ can be written in a matrix form.

$$\begin{bmatrix} x_{00} & x_{01} & \cdots & x_{0m} \\ x_{10} & x_{11} & \cdots & x_{1m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n0} & x_{n1} & \cdots & x_{nm} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

$$X\mathbf{a} = \mathbf{y}$$

$X_{n\times m}$ is not a square matrix and hence not invertible.

$X_{n\times m}$ Known

$\mathbf{a}_{m\times 1}$ Unknown

$\mathbf{y}_{n\times 1}$ Known

Least Squares Solution:

$$X^T X\mathbf{a} = X^T\mathbf{y} \quad \Rightarrow \quad \mathbf{a} = (X^T X)^{-1} X^T \mathbf{y} \qquad X^+ = (X^T X)^{-1} X^T$$

(Pseudo Inverse)

$$\boxed{\mathbf{a} = X^+\mathbf{y}}$$

Active Contours (Snakes)

What is an Active Contour?

Definition: a tool for finding the boundaries of objects in an image.

Task: The contour should evolve over time so that it latches on to the boundary of the object.

Image

What is an Active Contour?

Definition: a tool for finding the boundaries of objects in an image.

Task: The contour should evolve over time so that it latches on to the boundary of the object.

Active Contours (Also called Snakes):

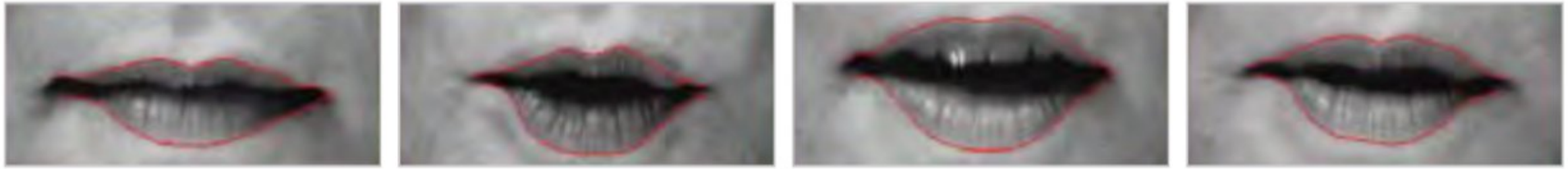
Iteratively "**deform**" the initial contour so that:

- It is near pixels with high gradient (edges)
- It is smooth

Image

Power of Deformable Contours

Boundaries could deform over time



Use case: Track objects over a video sequence.

e.g.: An active contour is first used to find the outline of the object (lips) in the first frame of the sequence. The contour for this frame can be used as the initial contour for the next frame

Power of Deformable Contours

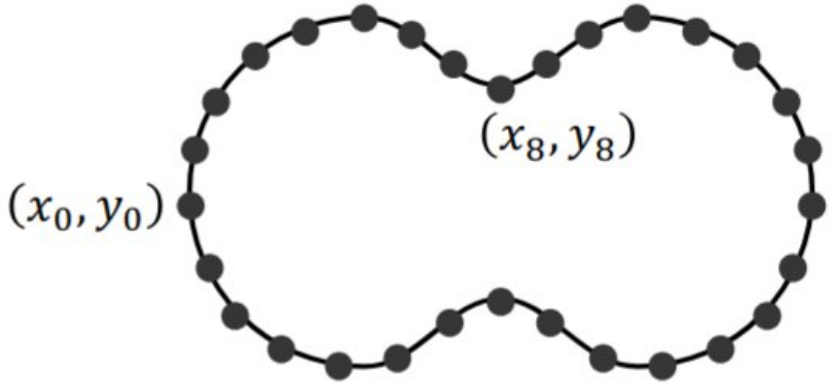
Boundaries could deform with

Use case: Track objects over a video sequence.

e.g.: An active contour is used to robustly find the boundary of the object even as its perspective changes with time.

Representing a Contour

Contour $\mathbf{v}$: An ordered list of 2D vertices (set of n points), called the **control points**, connected by straight lines of fixed length.



$$\mathbf{v} = \{v_i = (x_i, y_i) \mid i = 0, 1, 2, \ldots, n-1\}$$

Note: The sampling of the contour is uniform, such that the distance between consecutive control points is fixed.

Active Contour Initialization

Goal: Detect the boundary of a coin using an active contour (snake).

- A user-defined initial contour (blue dotted curve) is placed near the object.
- The contour must move and deform toward the true boundary.
- This movement is achieved by applying forces to the contour.

Image with
Initial Contour

Using Image Gradients

Gradients Indicate Object Boundaries

- Object boundaries usually have large image gradients.
- The gradient magnitude squared highlights strong edges in the image.
- These gradients can be used to generate forces that attract the contour toward the boundary.

Gradient
Magnitude Squared

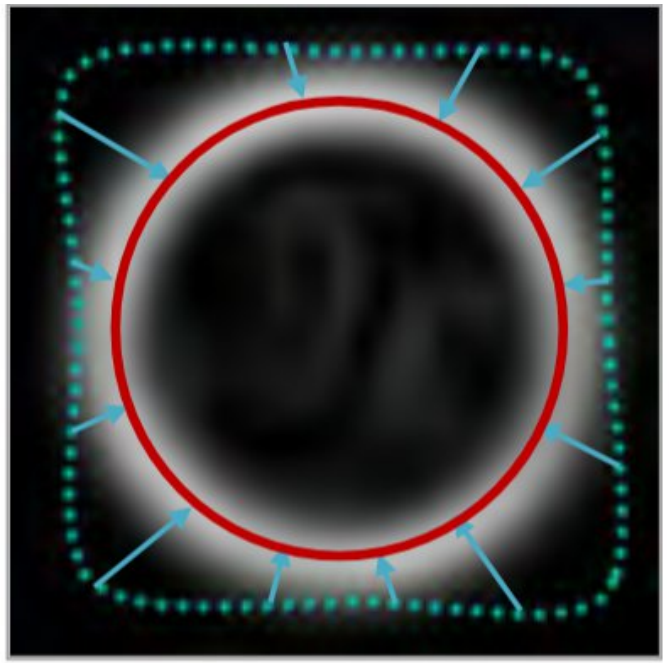
$$\|\nabla I\|^2$$

Problem: How do we attract the contour closer to the object when the gradients are sitting so far away?

Gradient Smoothing

Extending the Influence of Edges

- Trick: Smooth the gradient field using a Gaussian filter (σ).
- Smoothing spreads edge influence so distant contour points feel the force.
- The result acts as a force (potential) field guiding the contour.



Blurred Gradient Magnitude Squared

$$\|\nabla n_\sigma * I\|^2$$

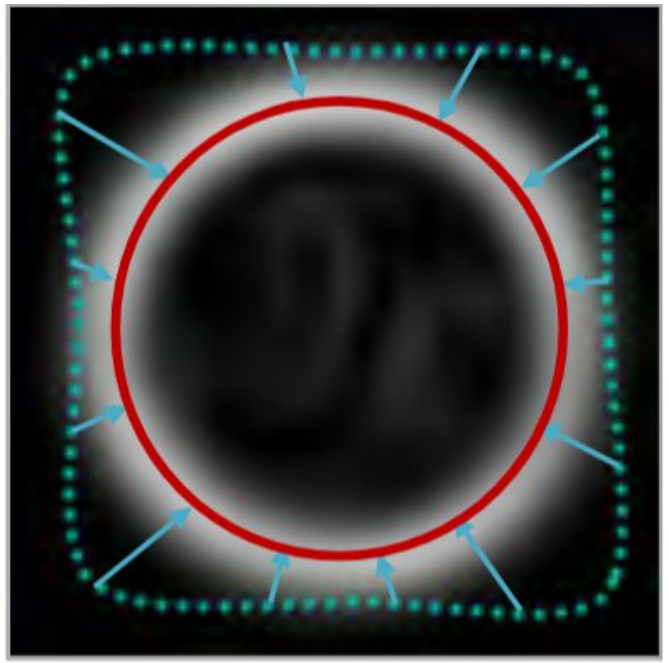
Optimization of the Contour

Goal: We want the contour to **maximize** the sum of gradient magnitudes along its points.

- If the contour lies on the boundary, the gradient values at its points will be large.
- If the contour is away from the boundary, the gradient values will be small.

So we compute:

$$\sum_{i=0}^{n-1} \|\nabla n_\sigma * I(v_i)\|^2$$



Blurred Gradient Magnitude Squared

$\|\nabla n_\sigma * I\|^2$

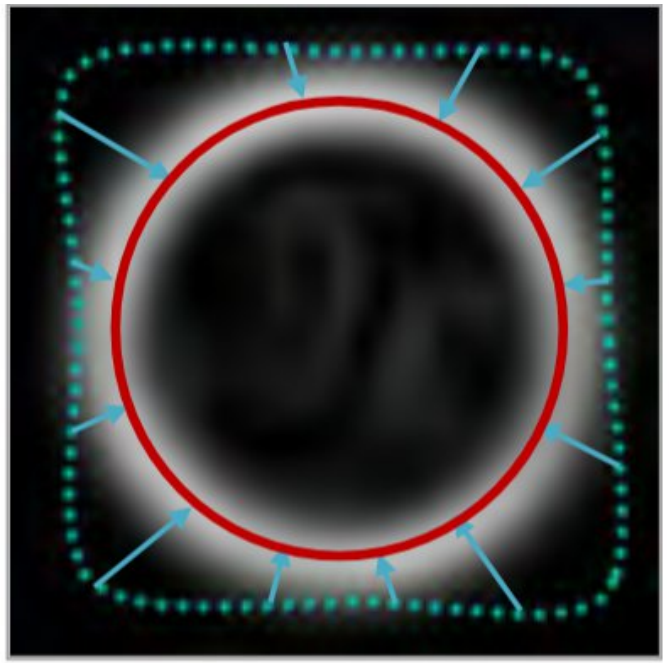
Optimization of the Contour

Instead of maximizing, we minimize the negative value.

So we define an energy (cost) function:

$$E_{image} = -\sum_{i=0}^{n-1} \|\nabla n_\sigma * I(v_i)\|^2$$

Minimizing this is equivalent to **maximizing** the gradient magnitude.



Blurred Gradient Magnitude Squared

$\|\nabla n_\sigma * I\|^2$

Contour Deformation: Greedy Algorithm

Goal: We want to **minimize the image energy** so the contour moves toward edges.

Each contour **point** v_i **moves** within a small search **window W** around its current position.

The point is **moved** to the location where E is **minimum**.

The window W

Contour Deformation: Procedure

Algorithm Steps:

1. For each contour point v_i :
   - Search within a **small window W** around the point.
   - Move v_i to the **position with minimum** E_{image}
2. Repeat this sequentially for all contour points.
3. Iterate the process until the total displacement of contour points is smaller than the **threshold**.

4. When the change is small → **stop the algorithm**.

Contour Deformation: Limitations

Limitations:

- Performance degrades when the **image is noisy**.
- Noise causes **noisy gradients and force fields**.

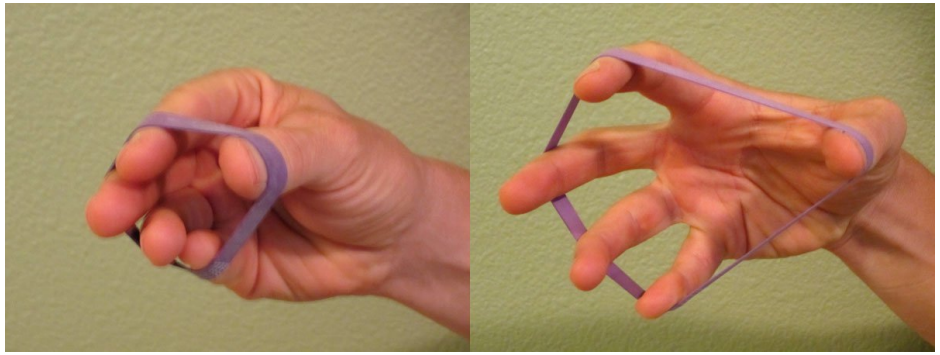
Result:

- The contour generally **reaches the object boundary**.
- However, it may contain irregularities or knots.

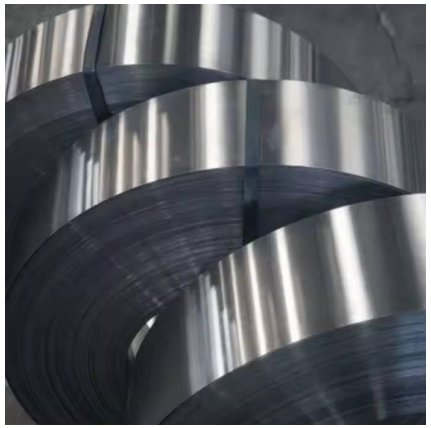
Adding Physical Constraints to the Contour

Making Contours Elastic and Smooth

1. The contour should behave like a physical object.
2. Two desirable properties:
   - **Elasticity** → behaves like a **rubber band**, allowing it to contract.
   - **Smoothness** → behaves like a **metal strip**, avoiding sharp bends.
3. These constraints prevent the contour from developing **irregular shapes** or **knots**.



Elasticity

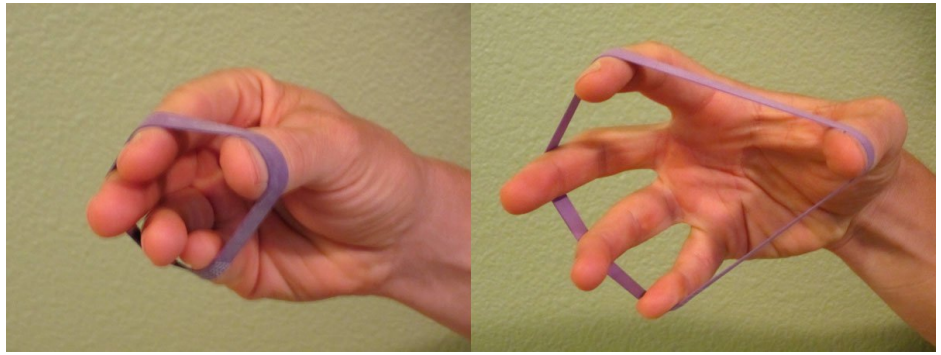


Smoothness

Internal Energy of the Contour

To enforce elasticity and smoothness, we define the internal contour energy as:

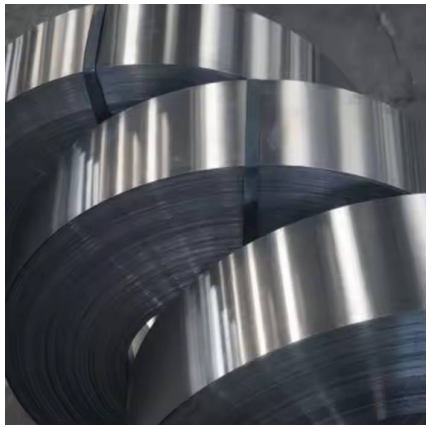
$$E_{contour} = \alpha E_{elastic} + \beta E_{smooth}$$



Elasticity

Where:

- $E_{elastic}$ → controls stretching or contraction of the contour.
- E_{smooth} → penalizes sharp bends, enforcing smoothness.
- α → weight controlling elasticity.
- β → weight controlling smoothness.



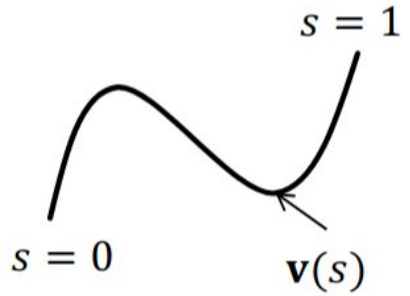
Smoothness

Formulating Elasticity and Smoothness

For point $0 \leq s \leq 1$ on continuous contour $\mathbf{v}(s) = (x(s), y(s))$:

$$E_{elastic} = \left\| \frac{d\mathbf{v}}{ds} \right\|^2$$

$$E_{smooth} = \left\| \frac{d^2\mathbf{v}}{ds^2} \right\|^2$$



Elasticity:

- Controls how much the contour stretches or contracts. We minimize the rate of change of the contour.
- To model elasticity, we want to penalize large changes in position, so the contour contracts or resists stretching.

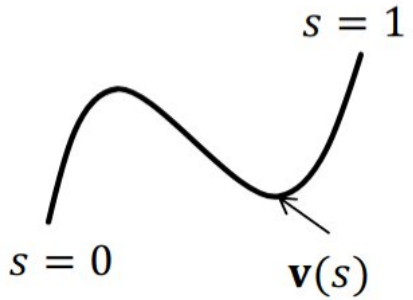
Smoothness:

- Controls how much the contour bends. Maximizing the smoothness means minimizing the curvature of the contour.
- Minimizing the magnitude of the second derivative reduces sharp bends, making the contour smooth.

Discrete Approximation (Control Points)

For point $0 \le s \le 1$ on continuous contour $\mathbf{v}(s) = (x(s), y(s))$:

$$E_{elastic} = \left\| \frac{d\mathbf{v}}{ds} \right\|^2 \qquad E_{smooth} = \left\| \frac{d^2\mathbf{v}}{ds^2} \right\|^2$$



Discrete approximations at control point $\mathbf{v}_i$:

$$E_{elastic}(\mathbf{v}_i) = \left\| \frac{d\mathbf{v}}{ds} \right\|^2 \approx \|\mathbf{v}_{i+1} - \mathbf{v}_i\|^2 = (x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2$$

$$E_{smooth}(\mathbf{v}_i) = \left\| \frac{d^2\mathbf{v}}{ds^2} \right\|^2 \approx \|(\mathbf{v}_{i+1} - \mathbf{v}_i) - (\mathbf{v}_i - \mathbf{v}_{i-1})\|^2$$

$$= (x_{i+1} - 2x_i + x_{i-1})^2 + (y_{i+1} - 2y_i + y_{i-1})^2$$

Elasticity and Smoothness

Internal bending energy along the entire contour:

$$\boxed{E_{contour} = \alpha E_{elastic} + \beta E_{smooth}}$$

where:

$$E_{elastic} = \sum_{i=0}^{n-1} [(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2]$$

$$E_{smooth} = \sum_{i=0}^{n-1} [(x_{i+1} - 2x_i + x_{i-1})^2 + (y_{i+1} - 2y_i + y_{i-1})^2]$$

Combining the Forces: The Snake Model

Image Energy, E_{image}: Measure of how well the contour latches on to edges

Internal Energy, $E_{contour}$: Measure of elasticity and smoothness

Total Energy of Active Contour:

$$E_{snake} = E_{total} = E_{image} + E_{contour}$$

Minimize the Total Energy

Snake Deformation: Procedure

Algorithm Steps:

1. For each contour point v_i :
   - Search within a **small window W** around the point.
   - Move v_i to the **position with minimum** E_{total}
2. Repeat this sequentially for all contour points.
3. Iterate the process until the total displacement of contour points is smaller than the **threshold**.

4. When the change is small → **stop the algorithm**.

Boundary around two objects

Large α
(More like a rubber band)

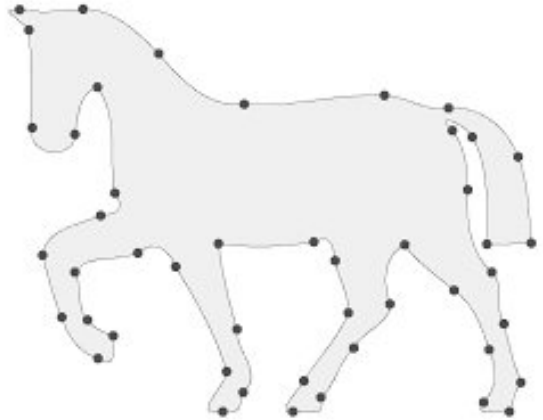
Small α
(Less like a rubber band)

Active Contours: Comments

Enhancing Active Contours with Shape Priors

- One method: introduce a shape prior term in the energy functional.
  - Penalizes deviations from a known object shape.

- Final contour now balances:
  1. Image forces (edges, gradients)
  2. Elastic forces (stretching/compression)
  3. Smoothness forces (avoiding sharp bends)
  4. Shape prior forces (matching a reference shape)



Benefit: More accurate segmentation when the object has a known or expected shape.

Active Contours: Comments

Why Initialization Matters in Active Contours

- Active contours are sensitive to starting positions.
- Poor initialization can lead to:
  1. Contours being too far from the actual object.
  2. Failure to form a proper force field that drives the contour correctly.
  3. Contours locking onto irrelevant structures in the image.

Benefit: Good initialization improves convergence and segmentation accuracy.

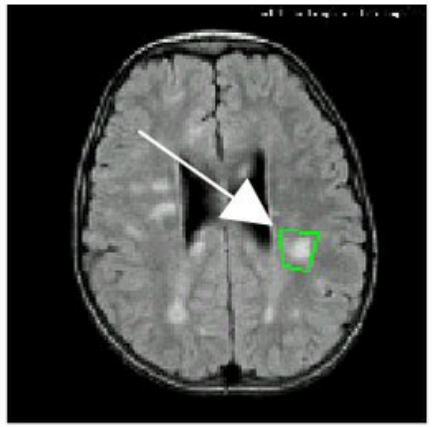
Active Contours: Comments

Contour Behavior: Contraction vs. Ballooning

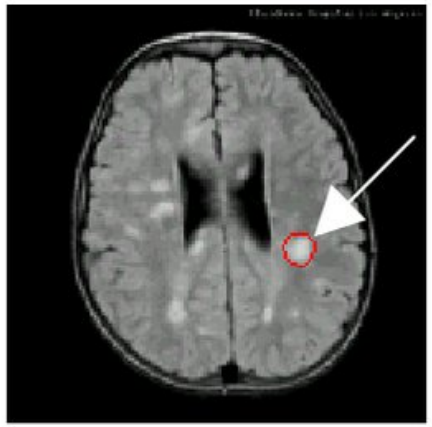
- Standard contour energy: forces cause contraction like a rubber band.
- Limitation: may not fully reach object boundaries if initialization is small.
- Alternative: ballooning forces that expand the contour outward.
  1. Start contour inside the object.
  2. Expand until it reaches the edges of the object.

Benefit: Can segment objects from inside-out, reducing dependency on precise edge initialization.

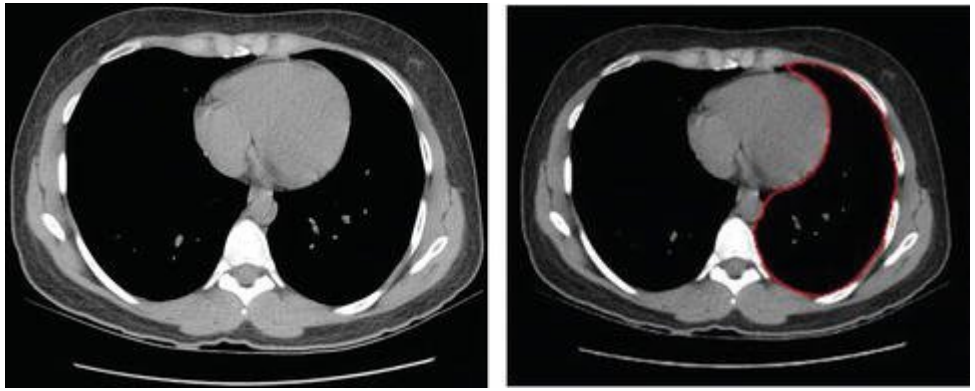
Medical Image Segmentation



(a) initial contour



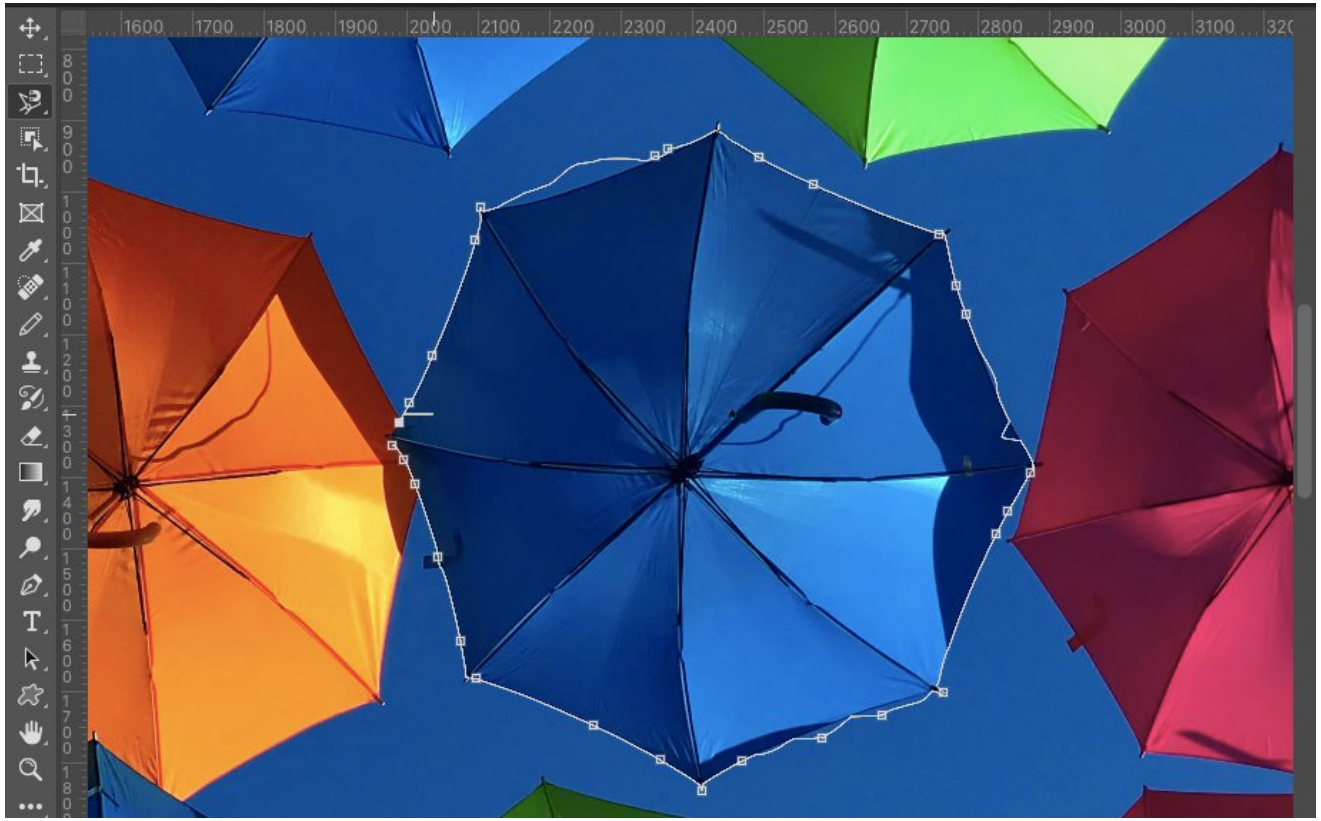
(b) final contour



Segmentation of chest image using snake model.

https://www.intechopen.com/chapters/59741

Interactive Image Segmentation



Magnetic Lasso Tool in Image Editing software